IN THE CLAIMS

1-8. (Canceled)

9. (Original) A processor comprising:

a decoder circuit to decode a load fence instruction;

a control register to enable a pre-serialization mode, a post-serialization mode, and an enhanced mode for said load fence instruction;

an execution unit to execute said load fence instruction after said decode circuit has decoded said load fence instruction.

10. (Original) The processor of claim 9 wherein said execution unit prevents load operations from executing until load operations executed prior to said load fence instruction are globally observed.

11. (Original) The processor of claim 10 wherein said execution unit blocks an instruction subsequent to said load fence instruction until said execution unit executes said load fence instruction.

12. (Original) The processor of claim 11 wherein said execution unit allows an instruction subsequent to said load fence instruction to execute out of order with respect to said load fence instruction if there is no dependency between said instruction and said load operations.

13. (Original) The processor of claim 12 further comprising a cache controller to control accesses to cache memory made in response to said execution unit executing a load operation.

14. (Original) The processor of claim 13 wherein said cache controller comprises a control register to control serialization of cache memory accesses made in response to executing said load operation.

15. (Currently Amended) A method comprising:

decoding a load fence instruction;

executing said load fence instruction after said decode circuit has decoded said load fence instruction;

enabling pre-serialization of operations appearing before the load fence instruction in program order and post-serialization of operations appearing after the load fence instruction in program order.

16. (Original) The method of claim 15 wherein said execution unit prevents load operations from executing until load operations executed prior to said load fence instruction are globally observed.

17. (Original) The method of claim 16 wherein a load buffer stores load operations to be executed by said execution unit.

18. (Original) The method of claim 17 wherein a reorder buffer stores load data resulting from executing said load operations.

19. (Original) The method of claim 18 wherein said execution unit blocks an instruction subsequent to said load fence instruction until said execution unit executes said load fence instruction.

20. (Original) The method of claim 19 wherein said execution unit allows an instruction subsequent to said load fence instruction to execute out of order with respect to said load fence instruction if there is no dependency between said instruction and said load operations.

21. (Original) The method of claim 20 further comprising a cache controller to control accesses to cache memory made in response to said execution unit executing a load operation.

22. (Original) The method of claim 21 wherein said cache controller comprises a control register to control serialization of cache memory accesses made in response to executing said load operation.

23-38. (Canceled)

39. (New) A system comprising:

a memory device comprising a first instruction, which if executed by a machine causes the machine to prevent a first load operation appearing after the first instruction in program order from executing until a second load operation appearing before the first instruction in program order is globally observed;

a processor comprising a control register to enable pre-serialization of the second operation and post-serialization of the first operation.

40. (New) The system of claim 39 wherein the processor comprises a decoder circuit to decode the first instruction and an execution unit to execute said first instruction.

41. (New) The system of claim 40 wherein said execution unit is to prevent the first load operation from being executed prior to the second load operation being globally observed.

42. (New) The system of claim 41 wherein said execution unit is to block a second instruction subsequent to said first instruction in program order until said execution unit executes said first instruction.

43. (New) The system of claim 42 wherein said execution unit is to execute a third instruction subsequent to said first instruction in program order out of order with respect to said first instruction if there is no dependency between said third instruction and said second load operation.

44. (New) The system of claim 43 further comprising a cache controller to control accesses to cache memory made in response to said execution unit executing a load operation.

45. (New) The system of claim 44 wherein said cache controller comprises a control register to control serialization of cache memory accesses made in response to executing a load operation.